7.11

(1) step 1

Consider the following starvation scenario:

1. There are five philosophers (P0, P1, P2, P3 and P4) who spend their lives thinking and eating.
2. They both P0 and P3 are allowed to eat first. P1, P2 and P4 attempt to eat, but are forced to wait. P0 and P3 finish at roughly the same time, but the scheduler chooses P1 and P2 to run when the conditions are signaled. Now, P0 and P3 both attempt to eat while P1 and P2 are eating and are told to wait. When P1 and P2 complete, the scheduler, which is unsympathetic, to the needs of P4, chooses P0 and P3 to run, and the cycle continues
3. P4 starves, because conditions arise that allow the scheduler to never choose it.

step 2

If every philosopher simultaneously picks up the left fork, then there will be no right fork to pick up. This will lead to starvation

(2) of Chapter 7-11

Step 1

To solve the starvation problem, the basic solution is that whenever a philosopher wants to eat, she checks both forks. If they are free, then she eats. Otherwise, she waits until a neighbor contacts her. Whenever a philosopher finishes eating, she checks to see if her neighbors want to eat and are waiting. If so, then she releases the fork to one of them and lets them eat.

Step 2

The difficulty is to first be able to obtain both forks without another philosopher interrupting the transition between checking and acquisition. We can implement this a number of ways, but a simple way is to accept requests for forks in a centralized priority queue, and give out forks based on the priority defined by being closest to the head of the queue. This provides both deadlock prevention and fairness.

3. from Chapter 7-11

step 1

When a philosopher makes a request for forks, allocate the request if one of the following conditions is satisfied:

1. The philosopher has two forks and there is at least one fork remaining.
2. The philosopher has one fork and there is at least two forks remaining
3. There is at least one fork remaining and there is at least one philosopher with three forks.
4. The philosopher has no fork, there are two forks remaining and there is at least one other philosopher with two forks assigned

Therefore where the request of the head of the queue does not have the closest forks available, though there are forks available for other philosophers.

4 from Chapter 7. 11

Step 1

By periodically repeating the request, the request will move to the head of the queue. Th only partially solves the problem unless we can guarantee that all philosophers eat for exactly the same amount of time and can use this time to schedule the issuance of the repeated request